Cypress Ref: CD04010
BSTZ Ref: 016820.P296

APPLICATION FOR UNITED STATES LETTERS PATENT

FOR

Method and System for a Feed-Forward Encoder

Inventor: **Edward Grivna**

Prepared by:

BLAKELY SOKOLOFF TAYLOR & ZAFMAN, LLP
12400 Wilshire Boulevard, 7th Floor
Los Angeles, California 90025
(503) 684-6200

Express Mail No: EV325530682US

Method and System for a Feed-Forward Encoder

BACKGROUND

1. Technical Field

[0001] Embodiments of the invention relate to the field of circuits for data communications encoding, and more specifically to a feed-forward encoder.

2. Background Information and Description of Related Art

[0002] A conventional 8B/10B encoder used for encoding signals for data communications is shown in Fig. 1. This encoder is described in U.S. Patent No. 4,486,739 and in an article entitled "A DC-Balanced Partitioned-Block, 8B/10B Transmission Code," IBM Journal of Research and Development, Volume 27, Number 5, September 1983, pages 440-451.

[0003] The encoder of Fig. 1 maps each incoming source 8-bit character 112 into a 10-bit encoded transmission character 118 based on the contents of the source character 112, a data/control selector bit 114, and a current running disparity. The running disparity of the character being encoded is calculated based on the resulting transmission character and is fed back to be used to encode the next character. The first five bits of each source character are fed into combinatorial logic block 102 that produces partial encodings based on the values of those bits. The last three bits are likewise fed into a similar block 104. These partial encodings are used, in conjunction with the current running disparity from the previous character, to determine how the final encoding should occur to maintain DC

balancing of the resulting transmission character stream. This is done by block 106, which uses the partial encodings determined by blocks 102 and 104 and the current running disparity of the previous transmission character. The final encoding is done in combinatorial logic blocks 108 and 110.

[0004] A conventional running disparity calculation function 106 is shown in Fig. 2. As shown, the six bits from the 5B functions block 102 and the current running disparity 210 of the previous transmission character are input into combinatorial logic block 202. Combinatorial logic block 202 resolves these seven bits to a single bit, which is then captured in flip-flop 214 on the falling edge of the clock 216. The output of flip-flop 214 tracks the running disparity of the first six bits of the transmission character, and is passed to logic blocks 204 and 206. The output of block 206 is used to control the final encoding of the last four bits of that transmission character. The four bits from the 3B functions block 104 and the running disparity 218 of the first six bits are input into combinatorial logic block 204. Combinatorial logic block 204 resolves these five bits to a single bit, which is then captured in flip-flop 212 on the rising edge of the clock. The output of flip-flop 212 is the running disparity of the entire transmission character, and is passed to logic block 208. The output of block 208 is used to control the final encoding of the first six bits of that transmission character. As shown by the cross-coupling in Fig. 2, it is necessary for one portion of the character to complete evaluation before the next section can begin the encoding process.

[0005] An alternative character encoder implementation 300 is shown in Fig. 3a. In this case, the same function shown in Figs. 1 and 2 is performed in a single clock

edge, instead of using a dual-phase clock. A large combinational logic block 302 is placed between the source character 304 and the output transmission character 306. The running disparity 308 is captured by flip-flop 310 once each character time, where it becomes the current running disparity 312 to be used in encoding the following character. The critical path in this implementation is the running disparity calculation of each character.

[0006] Due to the finite speed of logic circuits, it is often necessary to encode multiple characters in parallel. Fig. 3b illustrates a character encoder implementation that encodes multiple characters in parallel. The total time gained is effectively the number of characters encoded in parallel minus one, multiplied by the setup and clock-to-out time of the running disparity-tracking flip-flop. However, the delays in this implementation remain dominated by the sequential gating structures used to calculate the running disparity of each character.

[0007] Fig. 3c illustrates a character encoder implementation that speeds up a portion of the disparity calculation function when implemented across multiple characters. This implementation is described in U.S. Patent Application No. 60/503,570, filed on September 17, 2003, entitled "Faster 8B/10B Encoding and Decoding on Multi-Byte Datapath." In this implementation, one encoder for each character in the data path generates the new transmission character assuming the previous one ended with positive running disparity, and another encoder generates the new transmission character assuming the previous one ended with negative running disparity. A multiplexer is used to select one of each of the generated pairs of transmission characters. This allows the encode function to occur in parallel on

multiple characters and reduces the critical path to the delay of the running disparity calculation of the first character plus the delay through successive multiplexers of the remaining characters in the data path. However, this implementation requires the number of encoders to be doubled, the number of running disparity calculation circuits to be doubled, and adds multiple 11-bit-wide 2:1 multiplexers.

BRIEF DESCRIPTION OF DRAWINGS

[0008] The invention may best be understood by referring to the following description and accompanying drawings that are used to illustrate embodiments of the invention. In the drawings:

[0009] **FIG. 1** is a block diagram illustrating a conventional 8B/10B encoder.

[0010] **FIG. 2** is a block diagram illustrating a conventional running disparity calculation function.

[0011] **FIG. 3a** is a block diagram illustrating a conventional character encoder implementation.

[0012] **FIG. 3b** is a block diagram illustrating an alternate character encoder implementation with parallel encoding.

[0013] **FIG. 3c** is a block diagram illustrating an alternate character encoder implementation with reduced delays.

[0014] **FIG. 4** is a table of 8B/10B data characters.

[0015] **FIG. 5** is a diagram illustrating an exemplary decoder for the flip/hold function according to an embodiment of the invention.

[0016] **FIG. 6** is a diagram illustrating a flip/hold to running disparity converter according to an embodiment of the invention.

[0017] **FIG. 7** is a diagram illustrating a running disparity pre-calculator according to an embodiment of the invention.

[0018] **FIG. 8** is a flow diagram illustrating a method according to an embodiment of the invention.

DETAILED DESCRIPTION

[0019] Embodiments of a system and method for a feed-forward encoder are described. In the following description, numerous specific details are set forth. However, it is understood that embodiments of the invention may be practiced without these specific details. In other instances, well-known circuits, structures and techniques have not been shown in detail in order not to obscure the understanding of this description.

[0020] References throughout this description to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. Thus, the appearances of the phrases "in one embodiment" or "in an embodiment" in various places throughout this description are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0021] Embodiments of the invention describe an 8B/10B encoder which enables calculation of the running disparity of characters before the characters are encoded. In the conventional 8B/10B encoder 100 shown in Fig. 1, the running disparity is calculated based on the presently encoded character and fed back to be used to encode the next character. In the encoder of the present invention, the running disparity calculation block 106 is taken out of the conventional 8B/10B encoder 100, and the running disparity calculation is performed separately from the encoding of

the source characters. This enables the running disparity calculation to be performed prior to the encoding of the source characters.

[0022] Referring to Fig. 4, a table 400 of 8B/10B data characters is shown. This table is organized according to 5B and 3B partitioning of the data characters within the encoder and whether the characters maintain or invert the running disparity. An X indicates that the associated data character maintains the same running disparity, while a blank indicates that the associated data character inverts the running disparity. By analyzing the possible data characters in this way, one or more patterns may be recognized. For example, there are two types of columns, and each is the inverse of the other. There are also two types of rows, and each is the inverse of the other. Using these recognized patterns and a current running disparity value, the running disparity of one or more characters may be predicted prior to the encoding of the one or more characters.

[0023] Various circuits may be used to implement a decoder to determine whether a character will maintain (hold) or invert (flip) the running disparity. The logic for these decoders may be based on the table of Fig. 4 for data characters, and simplified using known methods, such as K-maps, Quine-McClusky, or Espresso. Fig. 5 illustrates one example of a decoder 500 for the flip/hold function according to an embodiment of the invention. This exemplary decoder covers all 256 possible data characters and takes as input a 9-bit character with eight data bits A to H and a data/control (D/K) selector bit. The A to H bits are each assigned specific binary weighting as defined in the IEEE 802.3 standard. The decoder 500 determines, based on the input data character, whether the character will maintain

or invert the current running disparity, relative to the previous character. If the character will maintain the current running disparity, the flip/hold bit 502 will be a logic value of 0. If the character will invert the current running disparity, the flip/hold bit 502 will be a logic value of 1. One or more circuit elements to handle the twelve control characters will also be implemented. The one or more circuit elements may be as simple as a single gate added to the decoder 500 of Fig. 5, depending on the control characters. The flip/hold bit 502 may be calculated at various locations in the data path, such as where the character is first generated or received, or just before the character is sent to an 8B/10B encoder. If the flip/hold bit is generated early in the data path, it can provide for error detection, since it contains redundant information based on the specific bit combinations present in each data character, similar to a parity bit.

[0024] Once the flip/hold bit is determined, the running disparity of each character may be determined by comparing the flip/hold bit with the current running disparity. In one embodiment, the current running disparity is Exclusive ORed (XORed) with the flip/hold bit. As a starting point, the current running disparity may be set to a specific value. For instance, certain protocols specify that running disparity should be initialized to a specific value. In this case, the current running disparity may be set to this value as a starting point to meet the requirements of the protocol. For example, in Fibre Channel, the standard specifies that following a reset, the running disparity should be initialized to negative. Therefore, for Fibre Channel, the current running disparity may be set to negative as a starting point to meet the requirements of the protocol. Alternatively, no specific value of running

disparity may be set in the beginning. In this case, the current running disparity may start out as being positive or negative, since no specific value is needed. In one embodiment, additional gates may be added to force specific running disparity using override characters.

[0025] Fig. 6 is a diagram illustrating a flip/hold to running disparity converter 600 according to an embodiment of the invention. The flip/hold bits, such as 620-626, are generated at some point in the data path using a decoder, such as the decoder 500 shown in Fig. 5, and are accepted as inputs along with the associated source characters, such as 630-636. These decoders may exist outside of the running disparity converter circuit 600, and the associated flip/hold bits are carried through the data path as part of the associated characters. Each flip/hold bit is compared using an XOR gate, such as 602-608, to a current running disparity, such as 662-668, to determine the running disparity for that character, and thus the current running disparity for the following character. The current running disparity for each character is then passed along with the character, such as 630-636, to an encoder, such as 610-616, to encode each character to a transmission character (TC). The output of the XOR gate 608, which is the running disparity 660 from the last character, is captured in a flip-flop 640 and becomes the current running disparity 662 on the following clock cycle.

[0026] In the circuit of Fig. 6, there is only a single feedback term that is based on a single XOR gate delay per character. Also, the encoders 610-616 do not have to contain any gates to calculate the running disparity of each character, since the running disparity of each data character is known before the character is encoded.

Therefore, the running disparity calculation logic may be removed from the conventional 8B/10B encoder 100 shown in Fig. 1, which simplifies the encoder logic significantly.

[0027] Fig. 7 illustrates a running disparity calculator 700 with partial pre-calculation according to an embodiment of the invention. The running disparity calculator 700 performs the same function as the circuit 600 shown in Fig. 6. However, the circuit 600 includes a cascade delay of a number of sequential XOR gates (four such gates are shown in Fig. 6). The running disparity calculator 700 reduces this delay path to a single 2:1 XOR gate (for example, gates 708-714 in Fig. 7) by recognizing that the XOR operations are fully commutative, or order independent. Therefore, most of the logic in the feedback term may be pre-calculated one or more clock cycles in advance, as shown in Fig. 7, using XOR gates such as 702-706 to determine an intermediate running disparity (ID), such as 770-774, based on the flip/hold bits 726-732. Each intermediate running disparity is then compared using an XOR gate, such as 708-714, to a current running disparity 782 to determine the running disparity of each character. The current running disparity 782 and the running disparities 784-788 are then passed along with the associated source characters 750-756 to encoders 716-722 to encode the characters to transmission characters. The output of the XOR gate 708, which is the running disparity 780 from the last character, is captured in a flip-flop 740 and becomes the current running disparity 782 on the following clock cycle.

[0028] The circuit 700 of Fig. 7 reduces the delay of the actual running disparity calculation to a single XOR gate, regardless of the number of characters operated

on in each clock cycle. Furthermore, the pre-calculation may be performed across multiple pipeline stages to reduce the delay even further. The pre-calculation of terms, performed for example by XOR gates 702-706, may be performed at numerous locations in the system and is not limited to the last pipeline register before the actual running disparity calculation is performed by gates 708-714.

[0029] Fig. 8 illustrates a method according to one embodiment of the invention. At 800, one or more source characters are evaluated to determine whether each character will invert or maintain a current running disparity. The source character may be a data character or a control character. In one embodiment, each character is evaluated to determine a flip/hold bit based on whether each character will invert or maintain a current running disparity. In one embodiment, the characters are evaluated using a lookup table to determine whether each character will invert or maintain a current running disparity. In one embodiment, the characters are evaluated using one or more logic gates to determine whether each character will invert or maintain a current running disparity.

[0030] At 802, a running disparity for each source character is determined before encoding the character based on the current running disparity and whether the character will invert or maintain the current running disparity. In one embodiment, the flip/hold bit of each character is compared with the current running disparity to determine a running disparity. In one embodiment, the flip/hold bit is compared with the current running disparity using an XOR gate. In one embodiment, the running disparity of each character is passed to an encoder to encode the following source character into a transmission character.

[0031] While the invention has been described in terms of several embodiments, those of ordinary skill in the art will recognize that the invention is not limited to the embodiments described, but can be practiced with modification and alteration within the spirit and scope of the appended claims. The description is thus to be regarded as illustrative instead of limiting.
